

## PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-212071

(43)Date of publication of application : 20.08.1996

(51)Int.Cl.

G06F 9/38

(21)Application number : 07-016189

(71)Applicant : MITSUBISHI ELECTRIC CORP

(22)Date of filing : 02.02.1995

(72)Inventor : ASAI TAKASHI  
KAWAMOTO KOJI

## (54) INSTRUCTION CODE GENERATION METHOD

## (57)Abstract:

PURPOSE: To provide an instruction code generation method which sets the value preceding by one cycle of a computing element in an input designation field of the computing element where the arithmetic result of one cycle is not stored.

CONSTITUTION: An instruction code (c) is generated based on a mnemonic code (a) and stored in an address A0. The code (a) designates to store the result of multiplication of both contents mx2 and my2 of an input register as the contents mz2 of an output register and also to store the result of addition of both contents az2 and my1 of the input register as the contents az2 of the output register. Then an instruction code (d) is generated in a cycle following that where the code (c) is carried out and stored in an address A1. The code (d) instructs an adder to store the result of addition of contents mx1 and my1 of both registers as the contents az2 of the output register and not to store the result of multiplication of contents mx2 and my2 of both input registers as the contents mz1 of the output register based on a mnemonic code (b) which designates to store the result of addition of contents mx1 and my1 of the input register as the contents az2 of the output register.

(a) mul mx2 my2 mz2 add az2 mx1 az2  
(b) add mx1 my1 az2

FPUX		48bit							
		EM	MY	MY	NZ	EN	MX	AY	AZ
(c)	A0	1	1	1	1	1	0	1	1
(d)	A1	1	0	1	1	1	1	0	1

## LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision]

of rejection]

[Date of requesting appeal against examiner's  
decision of rejection]

[Date of extinction of right]



## 【特許請求の範囲】

【請求項 1】 プロセッサの各单位サイクル毎に演算を行う演算器に対して、前記演算器の演算の結果をレジスタに格納しないことを指定するフィールドと、前記演算器に入力すべきデータを指定する入力指定フィールドを含む命令コードを生成する命令コード生成方法において、

前記入力指定フィールドに、生成すべき命令コードが実行される単位サイクルの 1 サイクル前に実行すべく生成した前サイクル命令コードの前記演算器に対応した入力指定フィールドの値を設定することを特徴とする命令コード生成方法。

【請求項 2】 前サイクル命令コードが 1 つである場合、該前サイクル命令コードの演算器に対応する入力指定フィールドの値を所定値とする請求項 1 記載の命令コード生成方法。

【請求項 3】 前サイクル命令コードが複数ある場合、該複数の前サイクル命令コードのいずれか 1 つに含まれ演算器に対応する入力指定フィールドの値を所定値とする請求項 1 記載の命令コード生成方法。

【請求項 4】 前サイクル命令コードが複数ある場合、該複数の前サイクル命令コードのうち実行回数が最も多い命令コードに含まれ演算器に対応する入力指定フィールドの値を所定値とする請求項 1 記載の命令コード生成方法。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】本発明は命令コード生成方法に関し、特に演算器の演算結果をレジスタに格納させない指定を含む命令コードを生成する命令コード生成方法に関する。

## 【0002】

【従来の技術】図 4 は演算器を備えた本発明が対象とするマイクロプロセッサの 1 例を示すブロック図である。図において 11, 12 はデータを転送するバスであって、バス 11 はデータを  $MX_1$  レジスタ 13 又は  $MX_2$  レジスタ 14 へ与え、バス 12 はデータを  $MY_1$  レジスタ 15 又は  $MY_2$  レジスタ 16 へ与える。 $MSEL_1$  セレクタ 17 は、両レジスタ 13, 14 のうちのいずれか一つを選択し、選択したレジスタの内容を MUL 乗算器 19 へ与える。 $MSEL_2$  セレクタ 18 は、両レジスタ 15, 16 のうちのいずれか一つを選択し、選択したレジスタの内容を MUL 乗算器 19 へ与える。MUL 乗算器 19 は両セレクタ 17, 18 から与えられる値を各サイクル毎に乗算し、乗算した結果を  $MZ_1$  レジスタ 20 又は  $MZ_2$  レジスタ 21 へ与える。

【0003】 $ASEL_1$  セレクタ 22 は、両レジスタ 13, 14 並びに  $AZ_1$  レジスタ 25 及び  $AZ_2$  レジスタ 26 のうちのいずれか一つを選択し、選択したレジスタの内容を ADD 加算器 24 へ与える。 $ASEL_2$  セレクタ 23 は、両レジスタ 20, 21 及び両レジスタ 15, 16 のうちのいずれか一つを選択し、選

択したレジスタの内容を ADD 加算器 24 へ与える。ADD 加算器 24 は、両セレクタ 22, 23 から与えられた値を各サイクル毎に加算し、加算した結果を両レジスタ 25, 26 のいずれか一つへ与える。

【0004】図 5 は図 4 のマイクロプロセッサを制御する命令コードの形式を説明する説明図であって、MUL 乗算器 19 及び ADD 加算器 24 の制御に関するものを示す。図において  $MX$  は  $MSEL_1$  セレクタ 17 が選択するレジスタを指定する 1 ビットのフィールドである。 $MX$  の値が “0” の場合に  $MX_1$  レジスタ 13 が指定され、 $MX$  の値が “1” の場合に  $MX_2$  レジスタ 14 が指定される。 $MY$  は  $MSEL_2$  セレクタ 18 が選択するレジスタを指定する 1 ビットのフィールドである。 $MY$  の値が “0” の場合に  $MY_1$  レジスタ 15 が指定され、 $MY$  の値が “1” の場合に  $MY_2$  レジスタ 16 が指定される。 $MZ$  は MUL 乗算器 19 が乗算した結果を格納するレジスタを指定する 1 ビットのフィールドである。 $MZ$  の値が “0” の場合に  $MZ_1$  レジスタ 20 が指定され、 $MZ$  の値が “1” の場合に  $MZ_2$  レジスタ 21 が指定される。 $EM$  は MUL 乗算器 19 が乗算した結果を格納するか否かを指定する 1 ビットのフィールドである。 $EM$  の値が “0” の場合に MUL 乗算器 19 の乗算結果はフィールド  $MZ$  の指定に関係なく両レジスタ 20, 21 のいずれにも格納されない。 $EM$  の値が “1” の場合に MUL 乗算器 19 の乗算結果はフィールド  $MZ$  で指定されたレジスタに格納される。即ち  $EM$  は乗算結果の格納の可否を指定するものであって、乗算の実行・非乗算を指定するものではない。

【0005】 $AX$  は  $ASEL_1$  セレクタ 22 が選択するレジスタを指定する 2 ビットのフィールドである。 $AX$  の値が “00” の場合に  $AZ_1$  レジスタ 25 が指定され、“01” の場合に  $AZ_2$  レジスタ 26 が指定され、“10” の場合に  $MX_1$  レジスタ 13 が指定され、“11” の場合に  $MX_2$  レジスタ 14 が指定される。 $AY$  は  $ASEL_2$  セレクタ 23 が選択するレジスタを指定する 2 ビットのフィールドである。 $AY$  の値が “00” の場合に  $MZ_1$  レジスタ 20 が指定され、“01” の場合に  $MZ_2$  レジスタ 21 が指定され、“10” の場合に  $MY_1$  レジスタ 15 が指定され、“11” の場合に  $MY_2$  レジスタ 16 が指定される。

【0006】 $AZ$  は ADD 加算器 24 が加算した結果を格納するレジスタを指定する 1 ビットのフィールドである。 $AZ$  の値が “0” の場合に  $AZ_1$  レジスタ 25 が指定され、 $AZ$  の値が “1” の場合に  $AZ_2$  レジスタ 26 が指定される。 $EA$  は ADD 加算器 24 が加算した結果を格納するか否かを指定する 1 ビットのフィールドである。 $EA$  の値が “0” の場合に ADD 加算器 24 の加算結果はフィールド  $AZ$  の指定に関係なく両レジスタ 25, 26 のいずれにも格納されない。 $EA$  の値が “1” の場合に ADD 加算器 24 の加算結果はフィールド  $AZ$  で指定されたレジスタに格納される。即ち  $EA$  は加算結果の格納の可否を指定するものであって、加算の実行・非実行を指定するものではない。

【0007】図 6 は図 4 のマイクロプロセッサを制御す

る命令コードで実行する操作を指定するニーモニッコードを説明する説明図である。図においてmul はMUL 乗算器19の乗算を制御するニーモニッコードであり、その後両セレクト17,18 が夫々選択するレジスタを指定するニーモニッコードが2つ及び乗算結果を格納するレジスタを指定するニーモニッコードが1つ随従する。このmul が記述されている場合、MUL 乗算器19の乗算結果は両レジスタ20,21 のいずれかに格納されるが、mul が記述されていない場合、両レジスタ20,21 のいずれにも格納されない。

【0008】add はADD 加算器24の加算を制御するニーモニッコードであり、その後両セレクト22,23 が夫々選択するレジスタを指定するニーモニッコードが2つ及び加算結果を格納するレジスタを指定するニーモニッコードが1つ随従する。このadd が記述されている場合、ADD 加算器24の加算結果は両レジスタ25,26 のいずれかに格納されるが、add が記述されていない場合、両レジスタ25,26 のいずれにも格納されない。mx1,mx2, my1,my2,mz1,mz2,az1,az2 は、夫々MX<sub>i</sub> レジスタ13, MX<sub>i</sub> レジスタ14, MY<sub>i</sub> レジスタ15, MY<sub>i</sub> レジスタ16, MZ<sub>i</sub> レジスタ20, MZ<sub>i</sub> レジスタ21, AZ<sub>i</sub> レジスタ25, AZ<sub>i</sub> レジスタ26を指定するニーモニッコードである。そして1行に記述されたニーモニッコードは1サイクルで実行する命令を指定する。例えば、1行にニーモニッコードmul とadd とが記述されている場合、2つの演算を同じサイクルで実行させることを意味する。

【0009】図6(a)はMX<sub>i</sub> レジスタ14の内容とMY<sub>i</sub> レジスタ16の内容とを乗算し、その乗算結果をMZ<sub>i</sub> レジスタ21に格納する操作とAZ<sub>i</sub> レジスタ26の内容とMY<sub>i</sub> レジスタ15の内容とを加算し、その加算結果をAZ<sub>i</sub> レジスタ26に格納する操作とを同時に行なうように指定するニーモニッコードである。図6(b)はMX<sub>i</sub> レジスタ13の内容とMY<sub>i</sub> レジスタ16の内容とを乗算し、その乗算結果をMZ<sub>i</sub> レジスタ20に格納する操作を指定するニーモニッコードである。図6(c)はMX<sub>i</sub> レジスタ13の内容とMY<sub>i</sub> レジスタ15の内容とを加算し、その加算結果をAZ<sub>i</sub> レジスタ26に格納する操作を指定するニーモニッコードである。マイクロプロセッサを制御する命令コードは図6で説明したニーモニッコードによる指定に従って生成される。

【0010】図7は図6に示すニーモニッコードの指定に従い、従来の命令コード生成方法で生成された命令コードの説明図である。図において(f)は命令コードを構成する各フィールドの記号を示し、(a)は図6(a)のニーモニッコードに対して生成された命令コードを示し、(b)は図6(b)のニーモニッコードに対して生成された命令コードを示し、(c)は図6(c)のニーモニッコードに対して生成された命令コードを示す。

【0011】従来の命令コード生成方法では、ニーモニッコードで指定されているフィールドにはその指定に

対応した値を設定し、ニーモニッコードで指定されていないフィールドには一定の値(以下デフォルトと称する)を設定する。あるサイクルにおいて乗算(又は加算)のみを行ない加算(又は乗算)を行わない場合、3個のフィールドAX,AY,AZ(又はMX,MY,MZ)にはデフォルトが設定される。従ってその場合両セレクト17,18(又は22,23)は夫々特定のレジスタを選択する。

【0012】図6(a)のニーモニッコードはMX<sub>i</sub> レジスタ14の内容とMY<sub>i</sub> レジスタ16の内容とを乗算し、乗算した結果をMZ<sub>i</sub> レジスタ21に格納し、同じサイクルでAZ<sub>i</sub> レジスタ26の内容とMY<sub>i</sub> レジスタ15の内容とを加算し、加算した結果をAZ<sub>i</sub> レジスタ26に格納すべきことを指定している故、生成した命令コードのEMフィールド、MXフィールド、MYフィールド、MZフィールド、EAフィールド、AZフィールドはいずれも“1”であり、AXフィールドは“01”であり、AYフィールドは“10”である。

【0013】図6(b)のニーモニッコードはMX<sub>i</sub> レジスタ13の内容とMY<sub>i</sub> レジスタ16の内容とを乗算し、乗算した結果をMZ<sub>i</sub> レジスタ21に格納すべきことを指定している故、生成した命令コードのEMフィールドは“1”、MXフィールドは“0”、MYフィールドは“1”、MZフィールドは“0”である。そして乗算のみを行ない加算を行わない故、加算結果は両レジスタ25,26のいずれにも格納しない。従って、EAフィールドは“0”である。デフォルトが“0”である場合、2桁の“0”は“00”となる故、AXフィールド、AYフィールドはいずれも“00”である。

【0014】図6(c)のニーモニッコードはMX<sub>i</sub> レジスタ13の内容とMY<sub>i</sub> レジスタ15の内容とを加算し、加算した結果をAZ<sub>i</sub> レジスタ26に格納すべきことを指定している故、生成した命令コードのEAフィールドは“1”、AXフィールドは“10”、AYフィールドは“10”、AZフィールドは“1”である。そして加算のみを行ない乗算を行わない故、乗算結果は両レジスタ20,21のいずれにも格納しない。従ってEMフィールドは“0”となる。デフォルトが“0”である場合、MXフィールド、MYフィールド、MZフィールドはいずれも“0”である。

【0015】

【発明が解決しようとする課題】必要としない演算結果は、いずれのレジスタにも格納されない故、必要としない演算を行なう演算器へ与えるデータはどのような値であっても構わなかった。それ故、デフォルトを設定することにより命令コードの生成を簡略化することができた。しかしながら、このような命令コードの生成方法は、前記演算器で消費される消費電力を低減するように考慮されたものではなかった。

【0016】前回のサイクルにおいて図7(a)に示される命令コードが実行され、今回のサイクルにおいて図7(b)に示される命令コードが実行される場合、MUL 乗算器19には前回のサイクルにおいて両レジスタ14,16の内

容が入力され、今回のサイクルにおいて両レジスタ13,16の内容が入力される。MSEL<sub>1</sub> セレクタ17は前回のサイクルにおいてMX<sub>1</sub> レジスタ14を選択し、今回のサイクルにおいてMX<sub>1</sub> レジスタ13を選択している。異なるレジスタの内容が同一である割合は、そのビット長と相関がある。例えば、レジスタのビット長が16ビットである場合、レジスタに格納されている値がランダムな値である場合、異なるレジスタの内容が同一である場合は1/65536である。

【0017】MSEL<sub>1</sub> セレクタ18は、前回及び今回の連続した両サイクルにおいてMY<sub>1</sub> レジスタ16を選択している。同一のレジスタの内容が両サイクルの期間において一定である割合は、その期間にレジスタに対して書き込みが行われる頻度と相関がある。例えば、1サイクルの間に4個のレジスタの中からランダムに選ばれた1個のレジスタにデータを書き込む場合、同一のレジスタの内容が両サイクルの期間において一定である割合は3/4である。

【0018】このように、前回及び今回の連続した両サイクルにおいて選択される両レジスタのうち一方が同じレジスタであり、他方が異なるレジスタである。そしてそれに伴いMUL 乗算器19に輸入されるデータが一定である割合は低くなる。従ってMUL 乗算器19で消費される電力の増加が見込まれる。しかしEMフィールドで乗算結果の格納が指定されているので、消費電力の増加は演算に伴う必然のものであり問題にはならない。同様にしてADD 加算器24で消費される電力のより一層の増加が見込まれる。これはEAフィールドで加算結果の格納が指定されていないので不要な電力消費であり問題である。この問題に対しAXフィールドに“01”を設定し、AYフィールドに“10”を設定するとよい。

【0019】また、前回のサイクルにおいて図7(a)に示される命令コードが実行され、今回のサイクルにおいて図7(c)に示される命令コードが実行される場合、MUL 乗算器19には、前回のサイクルにおいて両レジスタ14,16の内容が入力され、今回のサイクルにおいて両レジスタ13,15の内容が入力される。即ち前回及び今回の連続した両サイクルにおいて選択される両レジスタはいずれも異なるレジスタである。そして、それに伴いMUL 乗算器19に輸入されるデータが一定である割合は、より一層低くなる。従ってMUL 乗算器19で消費される電力のより一層の増加が見込まれる。そしてEMフィールドで乗算結果の格納が指定されていないので不要な電力消費であり、問題である。この問題に対し、MXフィールド及びMYフィールドにいずれも“1”を設定するとよい。同様にしてADD 加算器24で消費される電力の増加が見込まれる。これはEAフィールドで加算結果の格納が指定されているので消費電力の増加は問題にならない。

【0020】このように今回のサイクルにおいて演算結果の格納が指定されていない演算器がある場合、その演

算器にデータを入力すべきレジスタとして前回のサイクルにおいて選択したレジスタと同じレジスタを今回のサイクルにおいても選択する命令コードを生成することにより消費電力を節減できる命令コード生成方法を提供することを目的とする。

【0021】

【課題を解決するための手段】第1発明に係る命令コード生成方法は、プロセッサの各单位サイクル毎に演算を行う演算器に対して、前記演算器の演算の結果をレジスタに格納しないことを指定するフィールドと、前記演算器に入力すべきデータを指定する入力指定フィールドを含む命令コードを生成する命令コード生成方法において、前記入力指定フィールドに、生成すべき命令コードが実行される単位サイクルの1サイクル前に実行すべく生成した前サイクル命令コードの前記演算器に対応した入力指定フィールドの値を設定することを特徴とする。

【0022】第2発明に係る命令コード生成方法は、前サイクル命令コードが1つである場合、該前サイクル命令コードの演算器に対応する入力指定フィールドの値を所定値とすることを特徴とする。

【0023】第3発明に係る命令コード生成方法は、前サイクル命令コードが複数ある場合、該複数の前サイクル命令コードのいずれか1つに含まれ演算器に対応する入力指定フィールドの値を所定値とすることを特徴とする。

【0024】第4発明に係る命令コード生成方法は、前サイクル命令コードが複数ある場合、該複数の前サイクル命令コードのうち実行回数が最も多い命令コードに含まれ演算器に対応する入力指定フィールドの値を所定値とすることを特徴とする。

【0025】

【作用】第1発明では、演算器に入力すべきデータを指定する入力指定フィールドに、生成すべき命令コードが実行される単位サイクルの1サイクル前に実行すべく生成した前サイクル命令コードの前記演算器に対応した入力指定フィールドの値を設定する。従って生成した命令コードを実行する単位サイクルとその1サイクル前において、演算器に輸入されるデータが変化する割合が小さくなる。

【0026】第2発明では、第1発明において前サイクル命令コードが1つである場合該前サイクル命令コードの演算器に対応する入力指定フィールドの値を所定値とする。従って生成した命令コードを実行する単位サイクルとその1サイクル前において、演算器に輸入されるデータが変化する割合が小さくなる。

【0027】第3発明では、第1発明において、前サイクル命令コードが2つ以上である場合、その中のいずれか1つに含まれ、演算器に対応する入力指定フィールドの値を所定値とする。従って生成した命令コードを実行する単位サイクルとその1サイクル前において、演算

器に入力されるデータが変化する割合が小さくなる。

【0028】第4発明では、第1発明において、前サイクル命令コードが2つ以上ある場合、その中で実行回数が最も多い命令コードに含まれ、演算器に対応する入力指定フィールドの値を所定値とする。従って生成した命令コードを実行する単位サイクルとその1サイクル前とにおいて、演算器に入力されるデータが変化する割合が小さくなる。

【0029】

【実施例】以下本発明をその実施例を示す図面に基づき具体的に説明する。本発明に係る命令コード生成方法により生成される命令コードで制御されるマイクロプロセッサは、図4に示されるものであって、乗算器及び加算器を備え、1サイクルにおいて乗算及び加算を必要とする場合又は乗算若しくは加算を必要とする場合に演算を行なう。そして後者の場合において両演算器は演算を行なうが、必要としない演算を行った演算器の演算結果は格納されない。生成される命令コードの形式は図5に示されるものであって、乗算結果を格納するか否かを指定する1ビットのフィールドがEMであり、乗算すべきデータを格納するレジスタを指定する1ビットのフィールドがMX及びMYであり、乗算結果を格納するレジスタを指定する1ビットのフィールドがMZである。

【0030】また、加算結果を格納するか否かを指定する1ビットのフィールドがEAであり、加算すべきデータを格納するレジスタを指定する2ビットのフィールドがAX及びAYであり、加算結果を格納するレジスタを指定する1ビットのフィールドがAZである。生成すべき命令コードを指定するニーモニックコードは図6に示されるものであって、mul, add は夫々乗算、加算を制御するニーモニックコードであり、mx1, mx2, my1, my2, mz1, mz2, az1, az2 は夫々図4に示すレジスタMX<sub>1</sub>, MX<sub>2</sub>, MY<sub>1</sub>, MY<sub>2</sub>, MZ<sub>1</sub>, MZ<sub>2</sub>, AZ<sub>1</sub>, AZ<sub>2</sub>を示すニーモニックコードである。

【0031】図1は第1実施例に係る命令コード生成方法を説明する説明図である。図において(a)はアドレスA0に配置される命令コードを指定するニーモニックコードを示し、(b)はアドレスA1に配置される命令コードを指定するニーモニックコードを示し、(c)はアドレスA0に配置された命令コードの一部を示し、既に生成されているものであり、(d)は生成してアドレスA1に配置すべき命令コードの一部を示す。アドレスA0に配置された命令コードが実行されたサイクルの次のサイクルにアドレスA1に配置された命令コードが実行される。

【0032】図1(b)に示されるニーモニックコードはmulが記述されていない故、MUL乗算器19の乗算結果はレジスタに格納されない。従って図1(d)に示す命令コードを生成する場合、その命令コードの1サイクル前で実行される命令コード、即ち図1(c)に示される命令コードを調べる。そして、図1(c)のMXフィールドの値

1、MYフィールドの値1を図1(d)のMXフィールド、MYフィールドに設定する。その他のフィールドに対しては図1(b)のニーモニックコードで指定されているフィールドにはその指定に対応した値を設定し、ニーモニックコードで指定されていないフィールドにはデフォルトを設定する。即ち、MZフィールドにデフォルトとして“0”を、EAフィールドに加算結果の格納を示す“1”を、AXフィールドにMX<sub>1</sub>レジスタを示す“10”を、AYフィールドにMY<sub>1</sub>レジスタを示す“10”を、AZフィールドにAZ<sub>1</sub>レジスタを示す“1”を夫々設定し、アドレスA1に格納する。

【0033】図1(d)に示される命令コードは、こうして生成された命令コードである。このようにすることにより、両アドレスA0, A1に配置された命令コードを実行する2サイクルの期間においてMUL乗算器19に入力されるべく選択されるレジスタは、MX<sub>1</sub>レジスタ14、MY<sub>1</sub>レジスタ16となる。即ち同一のレジスタの内容が入力される故、MUL乗算器19で消費される電力は少なくなる。なお、本実施例においては、1サイクル前で実行される命令コードが1つ存在する場合について説明したが、1つも存在しない場合には図1(d)のMXフィールド、MYフィールドにデフォルトを設定する。また、割り込み処理ルーチンの先頭命令などのように、1サイクル前で実行される命令コードが予測できない場合は図1(d)のMXフィールド、MYフィールドにデフォルトを設定する。

【0034】図2は第2実施例に係る命令コード生成方法を説明する説明図である。図において(a)はアドレスA0に配置される命令コードを指定するニーモニックコードを示し、(b)はアドレスA1に配置される命令コードを指定するニーモニックコードを示し、(c)はアドレスA2に配置される命令コードを指定するニーモニックコードを示す。(d)は既に生成されてアドレスA0に配置された命令コードの一部を示し、図示しない他のフィールドにはシーケンス制御命令が含まれ、そのシーケンス制御命令によりアドレスA0に配置された命令コードの次にアドレスA2に配置された命令コードが実行される。

【0035】(e)は既に生成されてアドレスA1に配置された命令コードの一部を示し、図示しない他のフィールドにはシーケンス制御命令が含まれ、そのシーケンス制御命令によりアドレスA1に配置された命令コードの次にアドレスA2に配置された命令コードが実行される。(f)は生成してアドレスA2に配置すべき命令コードの一部を示す。

【0036】図2(c)に示されるニーモニックコードはmulが記述されていない故、MUL乗算器19の乗算結果はレジスタに格納されない。従って図2(f)に示す命令コードを生成する場合、その命令コードの1サイクル前で実行される命令コードを調べる。1サイクル前で実行される命令コードは図2(d)及び図2(e)に示される2個の命令コードである。このように1サイクル前で実行さ

れる命令コードが複数個存在する場合、そのうちの任意の一つの命令コードのMXフィールド、MYフィールドの値を生成すべき命令コードのMXフィールド、MYフィールドに設定する。図2(f)の場合には図2(d)の命令コードの両フィールドMX,MYの値(1,1)を設定している。

【0037】その他のフィールドに対しては図2(c)のニーモニックコードで指定されているフィールドには、その指定に対応した値を設定し、ニーモニックコードで指定されていないフィールドにはデフォルトを設定する。即ち、MZフィールドにはデフォルトとして“0”を、EAフィールドに加算結果の格納を示す“1”を、AXフィールドにAZ<sub>1</sub>レジスタを示す“01”を、AYフィールドにMY<sub>1</sub>レジスタを示す“10”を、AZフィールドにAZ<sub>2</sub>レジスタを示す“1”を夫々設定し、アドレスA2に格納する。

【0038】このようにすることにより、両アドレスA0,A2に配置された命令コードを実行する2サイクルの期間においてMUL乗算器19に☐入力させるべく選択されるレジスタはMX<sub>1</sub>レジスタ14、MY<sub>1</sub>レジスタ16となる。また両アドレスA1,A2に配置された命令コードを実行する期間においてMUL乗算器19に☐入力させるべく選択されるレジスタは先のサイクルにおいてMX<sub>1</sub>レジスタ13、MY<sub>1</sub>レジスタ16となり、後のサイクルにおいてMX<sub>1</sub>レジスタ14、MY<sub>1</sub>レジスタ16となる。従来のデフォルトを設定する場合は、後のサイクルにおいてMX<sub>1</sub>レジスタ13、MY<sub>1</sub>レジスタ15が選択される故、本実施例の方が両サイクルにおいてレジスタの種類が変化する割合が小さく、MUL乗算器19で消費される電力は少なくなる。即ち、乗算器の入力選択フィールドにデフォルトの値を設定する方式を用いた場合は、図6(d)の命令コードの後に図6(f)の命令コードが実行される時と図6(e)の命令コードの後に図6(f)の命令コードが実行される時のいずれの時も、MUL乗算器19に☐入力されるレジスタの種類が変化する場合も生じるので、本発明の方式の方が入力されるレジスタの種類が変化する割合が少ない。

【0039】図3は第3実施例に係る命令コード生成方法を説明する説明図である。図において(f)は生成してアドレスA2に配置すべき命令コードの一部を示す。(a),(b)…(e)は図2と同様であるので説明を省略する。図3(f)に示す命令コードを生成する場合、その命令コードの1サイクル前で実行される命令コードを調べる。1サイクル前で実行される命令コードは図3(d)及び図3(e)に示される2個の命令コードである。

【0040】図3(d)に示される命令コードはN<sub>1</sub>回実行され、図3(e)に示される命令コードはN<sub>2</sub>回実行され、N<sub>1</sub><N<sub>2</sub>である場合、図3(e)の命令コードのMXフィールド、MYフィールドの値(0,1)を生成すべき命令コードのMXフィールド、MYフィールドに設定する。図3(f)に示される命令コードは、こうして生成された命令コードである。このように1サイクル前で実行され

る命令コードが複数個存在する場合、その1サイクル前で実行される回数が最も多い命令コードの両フィールドMX,MYの値を生成すべき命令コードの両フィールドMX,MYに設定する。1サイクル前で実行される回数が最も多い命令コードが複数ある場合は、その中のいずれか一つの命令コードの両フィールドMX,MYの値を、生成すべき命令コードの両フィールドMX,MYに設定する。このようにすることにより、1サイクル前と現サイクルとにおいてMUL乗算器19に☐入力させるべく選択されたレジスタの種類が変化する割合が小さく、MUL乗算器19で消費される電力は少なくなる。

【0041】

【発明の効果】第1発明によれば、演算器に対応する生成すべき入力指定フィールドに、生成すべき命令コードが実行される単位サイクルの1サイクル前に実行すべく生成した前サイクル命令コードの前記演算器に対応した入力指定フィールドの値を設定する故、演算器に☐入力されるデータが変化する割合が小さくなり、演算器で消費される電力が少なくなる。

【0042】第2発明によれば、第1発明において前サイクル命令コードが1つである場合、その前サイクル命令コードの演算器に対応する入力指定フィールドの値を所定値とする故、演算器に☐入力されるデータが変化する割合が小さくなり、演算器で消費される電力が少なくなる。

【0043】第3発明によれば、第1発明において前サイクル命令コードが2つ以上である場合、その中のいずれか1つに含まれ、演算器に対応する入力指定フィールドの値を所定値とする故、演算器に☐入力されるデータが変化する割合が小さくなり、演算器で消費される電力が少なくなる。

【0044】第4発明によれば、第1発明において前サイクル命令コードが2つ以上ある場合、その中で実行回数が最も多い命令コードに含まれ、演算器に対応する入力指定フィールドの値を所定値とする故、演算器に☐入力されるデータが変化する割合が小さくなり、演算器で消費される電力が少なくなる。

【図面の簡単な説明】

【図1】 第1実施例に係る命令コード生成方法を説明する説明図である。

【図2】 第2実施例に係る命令コード生成方法を説明する説明図である。

【図3】 第3実施例に係る命令コード生成方法を説明する説明図である。

【図4】 マイクロプロセッサの1例を示すブロック図である。

【図5】 図4に示すマイクロプロセッサを制御する命令コードの形式を説明する説明図である。

【図6】 図4に示すマイクロプロセッサを制御する命令コードを指定するニーモニックコードを説明する説明

図である。

\*【符号の説明】

【図7】 従来の命令コード生成方法を説明する説明図である。

13,14,15,16,20,21,25,26 レジスタ、17,18,22,23  
\* セレクタ、19 乗算器、24 加算器。

【図1】

- (a) mul mx2 my2 mz2, add az2 my1 az2  
(b) add mx1 my1 az2

		命令コード							
アドレス		EM	MX	MY	MZ	EA	AX	AY	AZ
(c)	A0	1	1	1	1	1	01	10	1
(d)	A1	0	1	1	0	1	10	10	1

【図2】

- (a) mul mx2 my2 mz2, add az2 my1 az2  
(b) mul mx1 my2 mz2, add az2 my1 az2  
(c) add az2 my1 az2

		命令コード							
アドレス		EM	MX	MY	MZ	EA	AX	AY	AZ
(d)	A0	1	1	1	1	1	01	10	1
(e)	A1	1	0	1	1	1	01	10	1
(f)	A2	0	1	1	0	1	01	10	1

【図3】

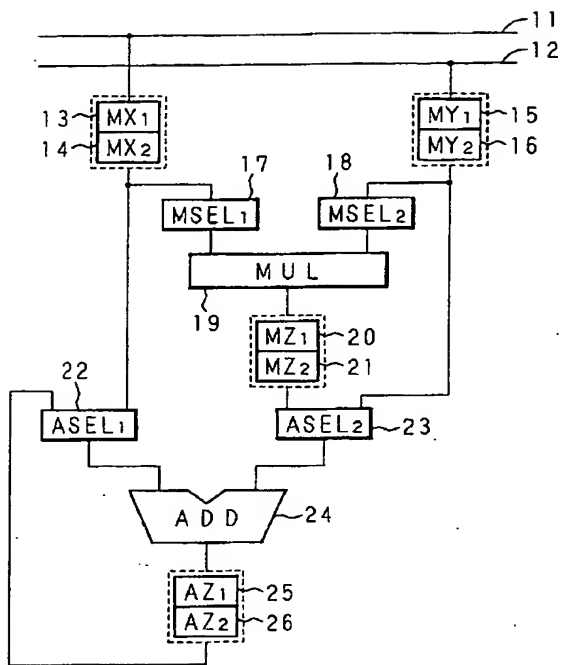
- (a) mul mx2 my2 mz2, add az2 my1 az2  
(b) mul mx1 my2 mz2, add az2 my1 az2  
(c) add az2 my1 az2

		命令コード							
アドレス		EM	MX	MY	MZ	EA	AX	AY	AZ
(d)	A0	1	1	1	1	1	01	10	1
(e)	A1	1	0	1	1	1	01	10	1
(f)	A2	0	0	1	0	1	01	10	1

【図5】

EM	MX	MY	MZ	EA	AX	AY	AZ
----	----	----	----	----	----	----	----

【図4】



11, 12: バス  
 13, 14, 15, 16, 20, 21, 25, 26: レジスタ  
 17, 18, 22, 23: セレクタ  
 19: 乗算器  
 24: 加算器

【図7】

(f)	EM	MX	MY	MZ	EA	AX	AY	AZ
(a)	1	1	1	1	1	01	10	1
(b)	1	0	1	0	0	00	00	0
(c)	0	0	0	0	1	10	10	1

【図6】

(a) mul mx2 my2 mz2, add az2 my1 az2  
 (b) mul mx1 my2 mz1  
 (c) add mx1 my1 az2